VZDU # 33 NOV/DEC 1991

VZ-EO DOWN UNDER CLUB

Editor
Harry Huggins
12 Thomas Str.
Mitcham. 3132
03-873-1408

Treasurer
Ron Allen
2 Orlando Str.
Hampton. 3188
03-598-4534

A MERRY CHRISTMAS TO ALL OUR CLUB MEMBERS, AND MAY YOU HAVE A VERY SUCCESSFUL NEW YEAR.

Now here is your christmas present. This issue is free. It does not count in your subscribed year! We are ahead with our finiances, so decided this was the better way to refund some.

I hasten to add that contributations used in this issue will receive the usual credit.

Inside you will find the rules for our design competition. There is also a sample of what can be done. I hope to see a big influx of entries,

This sample is really worth studying. There is very clever use of the READ-DATA routine that I have not often seen used. Not only the print characters but also the margins and line feeds are controlled by the READ-DATA routine. When first sighted I thought no hope for the VZ, but loading it, it did run. There is one statement had me puzzled in line 180......READY. I pondered on that a long time. Then the penny dropped. It was not READY but READ Y. When David Wood's series finishes, I shall go back to BASIC MADE EASY and explain this program. In the meantime see if you can follow it.

Our stand at the Box Hill SELL-SWAP was rather a flop. Only 2 people paid any attention to us. We were hoping to get some newmembers from it. A lot of our members have gone to IBM or the computers used at their schools.

So for this year I will say Cheerio. Let me have some input in the coming year.

Harry

DESIGN COMPETITION RULES

As I have not received any suggestions for this project, I have taken it upon myself to write them.

1. There will be only one class. There are a few will be barred. They are the 'experts'. They will know who I refer to. They have been contributing articles far in advance to what we hope to attain.

2. The competition is for a design that will reproduce on the VZ screen or printer. The program that produces it must be submitted so that others may do so. It may be for HI or LO resolution and for color or black and white. It must be done on a standard VZ. Extended Basic or Extended Dos may not be used.

3. Entries will be judged by all members attending the July 1992 meeting, acting as a committee. Their verdict will be final. Results will be in the JULY-AUGUST edition.

4. The prize will be 1 year's membership to the VZ DOWN UNDER CLUB.

5. All entries will be published over a period, and the author will receive the usual credit of one issue.

6. A statement that it is your own work must be provided.

In this issue is a sample. It may be an entry. I do not have a statement that it is his own work. Be that as it may it is well worth studying the programming style.

So, go to it and good luck.

BUNNY

```
  UN
 BUN                                          BUNNYB
 BUNNYB                                     NYBUNNYBUN
 BUNNYBUN                                 UNNYBUNNYBUN
  UNNYBUNNY                             NNYBUNNYBUNNYB
   NNYBUNNYBU                          UNNYBUNNYBUNNYB
    NYBUNNYBUNN                      YBUNNYBUNNYBUNNY
     YBUNNYBUNNY                   NNYBUNNYBUNNYBUNN
      BUNNYBUNNYB                 UNNYBUNNYBUNNYBUN
       UNNYBUNNYBU               BUNNYBUNNYBUNNYB
        NNYBUNNYBUN             YBUNNYBUNNYBUNNY
         NYBUNNYBUNNY          NYBUNNYBUNNYBUNN
          YBUNNYBUNNYB        NNYBUNNYBUNNYBU
           BUNNYBUNNYBU      UNNYBUNNYBUNNYB
            UNNYBUNNYBUN   BUNNYBUNNYBUNN
             NNYBUNNYBUN YBUNNYBUNNYBU
              NYBUNNYBUNNYBUNNYBUNNY
               YBUNNYBUNNYBUNNYBUNN
                BUNNYBUNNYBUNNYBU
                  NNYBUNNYBUNNY
                   NYBUNNYBUN
                    YBUNNYBU
                 UNNYBUNNYBUNN
              NYBUNNYBUNNYBUNNYB
            UNNYBUNNYBUNNYBUNNYBU
           BUNNYBUNNYBUNNYBUNNYBUN
         NYBUNNYBUNNYBUNNYBUNNYBUNN
        NNYBUNNYBUNNYBUNNYBUNNYBUNNY
       UNNYBUNN  UNNYBUNNYBUNNYBUNNY
      BUNNYBUN    UNNYBUNNYBUNNYBUNNYB
     YBUNNYBUN    UNNYBUNNYBUNNYBUNNYB
    NYBUNNYBUN  BUNNYBUNNYBUNNYBUNNYB
   NNYBUNNYBUNNYBUNNYBUNNYBUNNYBUNNYB
  UNNYBUNNYBUNNYBUNNYBUNNYBUNNYBUNNYB
   NNYBUNNYBUNNYBUNNYBUNNYBUNNYBUNNY
    NYBUNNYBUNNYBUNNYBUNNYBUNNYBUNNY
     YBUNNYBUNNYBUNNYBUNNYBUNNYBUNN
       UNNYBUNNYBUNNYBUNNYBUNNYBUNN
           BUNNYBUNNYBUNNYBUNNYBUN Y
               YBUN YBUNNYB  NYBU   B
                BUNNY   NYBUNNYB     U
               YBUNN  U  YBUNNYB      N
              NYBUNN     NYBUNNY    NYBUNN
             NNYBUNNYBUNNYBUNNY UNN
            UNN    N Y  N YBUNNYBU
           BU      NN    N Y    Y
                         NN  UNNY
                              NNY
                              ....
```

```
5 CLS:
10 LPRINT "                BUNNY
30 LPRINT:LPRINT:LPRINT
120 FOR I=0TO4:READB(I):NEXTI
130 GOSUB260
140 L=64
160 LPRINT
170 READX:IFX<0THEN160
175 IFX>128THEN240
180 LPRINT TAB(X);:READY
190 FORI=XTOY:J=I-5*INT(I/5)
200 LPRINTCHR$(L+B(J));
210 NEXT I
220 GOTO 170
240 GOSUB260:GOTO450
260 FORI=1TO6:LPRINTCHR$(10);:NEXT I
270 RETURN
290 DATA2,21,14,14,25
300 DATA1,2,-1,0,2,45,50,-1,0,5,43,52,-1
,0,7,41,52,-1
310 DATA1,9,37,50,-1,2,11,36,50,-1,3,13,
34,49,-1,4,14,32,48,-1
320 DATA5,15,31,47,-1,6,16,30,45,-1,7,17
,29,44,-1,8,19,28,43,-1
330 DATA9,20,27,41,-1,10,21,26,40,-1,11,
22,25,38,-1,12,22,24,36
335 DATA-1
340 DATA13,34,-1,14,33,-1,15,31,-1,17,29
,-1,18,27,-1
350 DATA19,26,-1,16,28,-1,13,30,-1,11,31
,-1,10,32,-1
360 DATA8,33,-1,7,34,-1,6,13,16,34,-1,5,
12,16,35,-1
370 DATA4,12,16,35,-1,3,12,15,35,-1,2,35
,-1,1,35,-1
380 DATA2,34,-1,3,34,-1,4,33,-1,6,33,-1,
10,32,34,34,-1
390 DATA14,17,19,25,28,31,35,35,-1,15,19
,23,30,36,36,-1
400 DATA14,18,21,21,24,30,37,37,-1,13,18
,23,29,33,38,-1
410 DATA12,29,31,33,-1,11,13,17,17,19,19
,22,22,24,31,-1
420 DATA10,11,17,18,22,22,24,24,29,29,-1
430 DATA22,23,26,29,-1,27,29,-1,28,29,-1
,4096
450 END
```

TO GET THE SAME PRINTOUT, ALTER THESE LINES AS THIS.
FOR EPSON COMPATIABLE PRINTERS.

```
5 LPRINTCHR$(27);CHR$(64);
7 LPRINT CHR$(27);CHR$(104);CHR$(1);
10 LPRINT "                BUNNY
20 LPRINT CHR$(27);CHR$(64);:LPRINTCHR$(27);CHR$(15);
25 LPRINTCHR$(27);CHR$(108);CHR$(10);
30 'LPRINT:LPRINT
```

ROBOTICS AND THE VZ

By BEN HOBSON and DAVE MAUNDER

In the last few months we have been experimenting with computer controlled vehicles. Not much is required in the form of electronics. The major difficulty is the constructing of a suitable vehicle. One way we built a vehicle was out of TECHNICAL LEGO using cogs to gear down the drive to tank like tread. The other way is to convert an old remote controlled car. These come in two types. One is controlled via a four way cable. The other is radio controlled. Either will do as long as you can change polarity to the motors with a double pole relay. In one of the prototypes, a radio controlled car was used. The transmitter was missing and as such the control circuitry was built into the actual car, with a twelve way cable running to the computer and power supply. As the circuit is so simple, construction details are left up to the individual.

The circuit is a modification of a circuit which appeared in the USBORNE book, 'PRACTICAL THINGS TO DO WITH A MICROCOMPUTER'. The original circuit did not have the capacitors of resistors, as it was designed for a BBC or similar computer which obviously has the 5v supply entirely seperate from the data lines. As such it did not work on the VZ. The resistors simply provide a current difference sufficient to turn on the transistors. The diodes and capacitors absorb any spikes produced by the switching of the relays. Without them, fast switching will at certain times hang the VZ. The single pole relay is the power on / off relay. It turns on or off power to the motors. This is necessary because the motor relays only reverse the polarity of the motors and such do not turn the motors off. In the circuit diagram the relays are represented by a little blob with a line either side. The blob is the common connection.

The bumper switches are two normally open micro switches. They are there to sense if your robot has hit anything. Connect them together via a strip of cardboard or tin to form a semi circular shape. This way any object in hitting the bumper will activate either or both switches. Two bumper circuits are shown. It is up to you which one to use if any. Two bumpers can be used, one at the front and one at the back.

Motors can be anything at all. Cassette motors appropriately geared are ideal. Just make sure they rotate at the same speed or the vehicle will veer to one side. The power supply can be either a battery or a plug pack transformer. Just don't try and use the VZ. If for some reason the motors stall, they will place an excessive load on the VZ's power supply and may damage it. Besides that it is doubtful if enough current can be supplied anyway. The VZ's regulator can only supply one amp and most of this is used up by memory expansions, disk controllers etc. To use the printer interface for the vehicle a small

modification is necessary. A 5v supply must be available at the centronics plug. Open up your printer interface and position it so the edge connector is forward and the PCB tracks are uppermost. Solder a wire to PIN 4 of the edge connector. This can be located by counting to the fourth pin from the left on the row of pins closest to the VZ side of the interface. To double check the pin behind should be connected to PIN 4 also. Although in older interfaces this may not be so. Now wind the wire around the cable, being sure to loop it through the straining clamp and connect it to a spare pin on the centronics plug. Centronics plugs have the pin numbers labelled to avoid confusion. In most cases PIN 36 is not used but you should check your printer manual to be sure. If pin 36 is used by the printer the 5v will disturb operation of the printer. However 99% of printers stick to the industrial standard layout and, according to that PIN 36 is not used. Reassemble the interface and check that 5v is available between PIN 36 (or your choice of pin) and PIN 16. If all is well you can build your vehicle.

The centronics pins and their functions are as follows

PIN 2 thru 9 - Data lines 0 thru 7
PIN 11 - Busy / ready
PIN 16 - Ground
PIN 36 - 5v

Writing software is easily achieved from BASIC. A simple OUT statement turns the data lines on or off. To calculate these values place a binary 1 in the appropriate data line number that corresponds to the relay you want turned on and convert this to decimal. Then type something like OUT 14,64. The port you send it to depends on your interface but try 12, 13, 14, 15.

To test your bumper you simply type

```
A=INP(0) : IF A=24 THEN GOSUB 1000 ELSE GOTO 20
```

The value for A can be discovered by the simple program.

```
10 A=INP(0)
20 PRINT A
30 GOTO 10
```

Press the bumper switches a few times and observe what the resulting number is.

That is all except for a sample program. It can be expanded to however you wish but each program depends on the individual vehicle.

GAMES COLUMN

He's back again. (and only just made it. Ed.), and his exams have finished (over a week late Ed.), just in time for another VZDU games column. I've even hear from some people since the last edition.!!!! Peter Watson has once more quietly blitzed the High Scores chart, while still finding time to hit us with some electifying game hints. Thanks Peter. I also have to say Hello and thank you to Bernice O'Mahoney and Tim and Mitch Pendlebury for their high score contributions. Thanks Guys. VZ gaming is alive and kicking.

One thing that still remains bare though is this bit which I used to set aside for Adventure game reviews. Unfortunately it seems i've run out of adventures to review. Although I would be very interested to hear from any of David Wood's pupils from his adventure writing series. VZ adventure games need something new, and I'm sure there are many out there who are capable of writing their own stuff. I myself plan to spend some time over my long Christmas holiday revising David'sseries in an effort to create my own masterpiece. Let's see what we can come up with.

HOT HANDY AND HELPFUL TIPS

LUNAR LANDER. Don't land on a fuel pad until your fuel level drops to about 2000 or less. There are 3 pads marked 5000 points, but the left-hand one is actually a 3000 point pad that has been mis-labeled.---Peter Watson.

PLANET PATROL. Always press the FIRE button before jumping over any obstacle, as there is often another on the far side of the thing you jump over.--Peter Watson.

SPACE RAM. Just a small hint for those High Score Chasers. I've found pleanty of points can be made by knocking off a few ships at the start of each stage and then waiting for each remaining ship to turn into time bombs before shooting them. If you destroy the time bombs quickly hundreds of points can be gained from each stage.

QUESTIONS and QUERIES

???? NO QUESTIONS? Surely we are not all content with our VZ games???

HIGH SCORES

GAME	SCORE	LEVEL	HOLDER
DAWN PATROL	78100		Paul Frantz
CRASH	881		Matthew McLean
DIG OUT	52500		Kenley McLean
HAMBURGER SAM	51000		Roger McLean
LADDER CHALLENGE	23970		Peter Watson
KAMIKAZE	113410		Peter Watson
VZ INVADERS	30160		Peter Watson
GALAXON	328460		Matthew McLean
PENGUIN	2800		Matthew McLean
LUNAR LANDER	52520		Peter Watson
SUPER SNAKE	1918		Peter Watson
MAZE OF ARGON	78306		Peter Watson
ASTEROIDS	110000		Peter McLean
CIRCUS	3180		Matthew McLean
PANIK	11090		artin Wedgwood
HOPPY	25550		Matthew McLean
GHOST HUNTER	23400		Chris McLean
KNIGHTs & DRAGONS	5300	EASY	Peter Watson
KNIGHTS & DRAGONS	1200	EXPERT	Peter Watson
SPACE RAM	1441		Matthew McLean
MISSILE ATTACK	52000		Heru McLean
BUST OUT	2600		Peter Watson
PLANET PATROL	1091		Peter Watson
DEFENCE PEN.	1563		Peter Watson
PHAROAH'S CURSE	135	SHILL 5	Peter Watson
STAR BLASTER	480		Matthew McLean
We don't have the level for this score. I think would BE 5. Matt will verify it for the next edition. (Ed.)			
STAR BLASTER	787	LEVEL 1	Tim Pendelbury
STAR BLASTER	683	LEVEL 2	Tim Pendelbury
STAR BLASTER	625	LEVEL 3	Tim Pendelbury
STAR BLASTER	419	LEVEL 4	Tim Pendelbury
STAR BLASTER	219	LEVEL 5	Tim Pendelbury

Time up for me again. It's been brief but we are workig on making it bigger. I hope everyone has a great Xmas, and an even better New Year. The Xmas break should give us all a chance to get into some serious VZ gaming.! Well I hope so.

Anyhow as always, see you next edition.

Please send all your hints, high scores and questions and reviews in to:-

Paul Frantz, 25 Crocker St. KIRWAN. Q'ld., 4817.

ADVENTURE GAMES-LONGER ROOM DESCRIPTIONS

Some time ago I mentioned that if you wanted to have long room descriptions, you could use a method known as tokenising. This method stores the descriptions in DATA statements with many of the commonly used words instead replaced by a single token. The tokens used in our case will be inverse characters, which are the ASCII characters from 192 to 255. This leaves 63 characters for tokens, with a 64th character used for a different purpose which we will get to later.

However I then went on to say that programming this method was rather tedious, as you had to look for common words, assign a token for each word, and then type in each data statement, replacing words with the inverse tokens where this is required. I find that this task is much better left to a computer, so I have written a program to deal with this. (The program is based on the "Single Line Data Statement" program which appeared in the March 1987 edition of VZ User.)

The program won't discover commonly used words for you, so you will still have to do this yourself. As a rule of thumb, a word should occur at least three times before it is worth assigning it a token. Short words can be given tokens too, as the space following the word can also be removed. (I found that in the demonstration program, the biggest memory saving from a single word was from the string "the ".) Once you have picked out your tokens, you might like to re-word your room descriptions slightly, so that more tokens can be used.

When you run the program, it will ask you to enter the tokens. It doesn't matter what order you do this in, but I recommend you keep the words "A","AN","THE" and "SOME" together as you will need these to describe gettable objects. It will also be useful to keep "NORTH", "SOUTH", "WEST" and "EAST" together. The program will create a DATA line containing the words you enter.

You can then enter the room descriptions (in full, with the exception of the first two words - "You Are") and the program will search through the description word by word for tokens, and add data lines to itself containing the tokenised descriptions. Unless you want to use some other approach for the movement codes, you will need to place these at the end of each description. Each description will be on a single data line, despite the fact that many will be longer than sixty four characters. For this reason you will only be able to edit some of your descriptions (so it is important to make sure they are correct), but the reading of the room descriptions by the program will be made a lot simpler.

However, you will lose some of the flexability that you could have obtained by entering the data lines manually, as you will be unable to use tokens for parts of words. (For example if you had the token [inverse &] for the group of letters "AND ", then "SAND " could be represented by "S[inverse &]". For this reason you might want to enter your data lines manually, as I did when I originally wrote the demonstration program, but it should not be too difficult to modify the tokens program so it can also handle this possibility.

Another idea you might want to consider is to also tokenise the replies given by R$. These could be placed in a series of data lines placed after the room descriptions, and the verb subroutines could return a number representing that message. The program could then use the restore line number routine to read this message from the data statements before decoding and printing it. The tokenising program would need little or no modifications to accomodate this, but possibly the biggest advantage of this approach would be that players who list your program will have a hard time trying to cheat.

If you use the program to create your descriptions, then decoding them will be simple. If you don't use it, then this will be made slightly more complex because you will need to work out how many lines you will need to read in. The code for reading a single data line with the movement codes at the end of it is below: (It assumes that all of the words that have tokens have been read into the Z$ array.)

```
4800 READB$
4810 D$=RIGHT$(B$,4)
4815 FORI=1TOLEN(B$)-4:E$=MID$ (B$,I,1)
4820 IFE$<"@"THENJ$=J$+E$:NEXT: RETURN: REM @ IS INVERSED
4830 IFE$="?",J$=LEFT$(J$,LEN (J$)-1)+". ":NEXT:RETURN:' INVERSE ?
4840 J$=J$+Z$(ASC(E$)-191): NEXT:RETURN
```

Here you will notice the use of the 64th token. Some tokens will occur at the end of a sentence, but when they are written to the screen they, like all of the other tokens, will be followed by a space. In this case, the space is undesirable, so we must cut it out of the description. To do this we use a special token (the inverse question mark) to instruct the computer to remove the preceeding space before printing a full stop and another space.

Because the program places each description on a single data line, there is no particular reason why the movement codes should be placed at the end of the description. An alternative way of representing the movement codes would be as a four bit number for each room. To work out each movement code, start off with it equal to zero, add 8 if there is an exit to the north, 4 if there is an exit the south, 2 if there is an exit to the west, and 1 if there is an exit to the east. Possible ways of storing the movement codes are in an array or group of data statements, or in a string. to do this, take each movement code number, add 32 so it is a printable ASCII character, look up what this character is in the VZ BASIC Reference Manual, and place it in a string in the same order as the room numbers. for example:

```
EX$= "$%"+ CHR$(34) +"%&,,$,,-/*,,-+#*))###" +CHR$(34)
```

To obtain the code you want:

```
EX = ASC( MID$(EX$,R,1)) -32
```

To decide if there is an exit or not:

```
860 EO = EX AND (2^(4-D))
870 IF EO>0 THEN R=R+VAL(MID$ ("-505-101",D*2-1,2)):R$="OK"
```

No, line 860 is not an error! Some VZers could be excused for not having heard of the AND function as it is not very well documented in some editions of the reference manual. The AND function compares the bits of two numbers, and if both numbers have a particular bit set (equal to one) then the result will also have that bit set.

```
eg. 13 AND 7
13    1101
 7    0111
      ----
 =    0101    (5)
```

Therefore 13 AND 7 equals 5.

If you still don't understand, consult an article on logic operators, such as the one by Bob Kitch which appeared in VZDU #20 and a number

of other VZ publications. Of course, you do not have to use this method in particular, but it is fairly difficult to decipher for those trying to cheat, and the exits can be altered quite easily:

EX$ = LEFT$(EX$, RM-1) + "(new character)" + MID$ (EX$, RM+1)

The program to tokenise your room descriptions is below:

```
10 CLEAR 2000:BS$=CHR$(8):RT$=CHR$(13):S=65536:E=33491-S
200 INPUT"HOW MANY TOKENS (1-63)";TK
210 IFTK<1 OR TK>63 THEN 200
220 DIM TK$(TK)
230 FOR T=1 TO TK
240     PRINT "TOKEN NUMBER";T;
250     INPUT TK$(T)
260 NEXT T
270 INPUT "DO YOU WANT A PRINTOUT (Y/N)";AN$
280 IF AN$="Y"THENGOSUB1000
285 GOSUB 1100
287 GOSUB3000
290 PRINT "TYPE YOUR ROOM DESCRIPTION"
300 GOSUB1500
310 CLS:PRINTDE$
320 PRINT:INPUT"IS THIS CORRECT (Y/N)";AN$
330 IF AN$="N" THEN 290
340 TD$=""
350 GOSUB2000
360 LN=LN+10: GOSUB3200
370 INPUT "ANOTHER DESCRIPTION";AN$
380 IF AN$<>"N" THEN CLS:GOTO290
390 POKE28672,INT((E+S)/256):POKE28673,E+S-256*PEEK(28672)
400 POKE30969,PEEK(28673):POKE30970,PEEK(28672)
990 STOP
1000 FOR T=1 TO TK
1010     LPRINT USING"##"; T;
1015     LPRINT " "; CHR$ (T+191);" "; TK$(T),
1020 NEXT
1030 RETURN
1100 INPUT "ARE THESE CORRECT NOW (Y/N)";AN$
1110 IF AN$<>"N" THENRETURN
1120 INPUT "WHICH ONE IS INCORRECT";IN
1130 PRINT "TOKEN NUMBER";IN;
1140 INPUT TK$(IN)
1150 GOTO 1100
1500 IN$=RT$:DE$=""
1510 PRINT"█";
1520 OL$=IN$:IN$=INKEY$:
1530 IF IN$="" OR IN$=OL$ THEN 1520
1540 IF (IN$<" " OR IN$>"^") AND IN$<>BS$ AND IN$<>RT$ THEN 1520
1560 IF IN$=BS$ AND LEN(DE$)<>0 THEN DE$=LEFT$(DE$,LEN(DE$)-1)
1570 IF IN$=BS$ THEN PRINT BS$; BS$; CHR$(127); "█";:GOTO1520
1580 IF IN$=RT$ OR LEN(DE$)=255 THEN 1650
1590 DE$=DE$+IN$: PRINT BS$; IN$; "█";
1600 GOTO1520
1650 PRINTBS$;" ":RETURN
2000 LS=1
2010 FOR LE=1 TO LEN(DE$)
```

```
2020     IFMID$(DE$,LE,1)=" "THEN GOSUB2500:LS=LE+1
2030 NEXT LE
2035 GOSUB2500
2040 PRINTTD$
2050 RETURN
2500 FS=0:WO$=MID$(DE$,LS,LE-LS)
2510 IFRIGHT$(WO$,1)="."THENFS=1:WO$=LEFT$(WO$,LEN(WO$)-1)
2515 TR=0
2520 FOR I=1 TO TK
2530     IF WO$=TK$(I) THEN TR=1:TD$=TD$+CHR$(191+I):I=TK
2540 NEXT
2550 IF TR=1 AND FS=1 THEN TD$=TD$+"▓"
2560 IFTR=0THENTD$=TD$+WO$:IFFS=1THENTD$=TD$+". "ELSETD$=TD$+" "
2570 RETURN
2700 MS=INT(LN/256):LS=LN-(MS*256):POKE E,LS: POKE E+1,MS
2710 POKE E+2,136: E=E+3: RETURN
3000 LN=4990:GOSUB2700
3010 FOR T=1 TO TK
3020     FOR L=1 TO LEN(TK$(T))
3030         POKE E,ASC(MID$(TK$(T),L,1))
3040         E=E+1
3050     NEXT L
3060     IF T<>TK THEN POKE E,44:E=E+1
3070 NEXT T
3080 POKE E,0: POKE E+1,0: POKE E+2,0: EL=E+1: E=E+3
3090 POKE33490-S,INT((E+S-2)/256)
3095 POKE33489-S,(E+S-2)-256*PEEK(33490-S)
3100 RETURN
3200 GOSUB2700
3210 POKE E,34: E=E+1
3220 FOR B=1 TO LEN(TD$)
3230     POKE E,ASC(MID$(TD$,B,1))
3240     E=E+1
3250 NEXT B
3260 POKE E,34:E=E+1
3270 POKE E,0: POKE E+1,0: POKE E+2,0: E=E+3
3280 POKE EL+1,INT((E+S-2)/256)
3295 POKE EL,(E+S-2)-256*PEEK(EL+1):EL=E-2
3300 RETURN
```

HOW TO USE THIS PROGRAM

Type in the above listing. You don't have to type it exactly as it appears, but you may find it helpful to do so. Once completed, check the End Of Basic (EOB) pointer by typing:
PRINT PEEK(30969) + 256 * PEEK (30970)

Replace the value of E in line 10 with E=(the value of EOB) - S. As well as this replace the number 33490 which occurs in lines 3090 and 3095 with the value of the EOB pointer MINUS one. replace the number 33489 in line 3095 with the value of the EOB pointer minus two.

Now save your program, because if it has any bugs it may crash. After you have done so, you must set aside some memory for your tokens and for your room descriptions. You can be fairly generous with this as the program resets the EOB pointer to the new correct value once it has finished. to do this type:
POKE30970, PEEK(30970) + n
this will allocate (n times 256) BYTES of memory, so if n equals eight, you have set aside 2K of memory.

Work out which words you want tokenised then run the program. type the number of tokens you want to use, then each token one by one. If you request a printout, the program will dump each word and its token to a printer. (you will need a printer patch loaded.) If you made a mistake typing one of the words you can correct it by typing the number of the incorrect word, then the correct spelling.

Now type your first room description. This doesn't use the normal INPUT. to edit the left arrow moves the cursor left AND rubs out. Don't worry if a black blob is left in the description - they have no effect. Your description must be no longer than 255 characters. You will then be asked if the description is correct - if it isn't you may type it again. After this the description with words replaced by tokens is printed on the screen, then you will be prompted if you want to type another description.

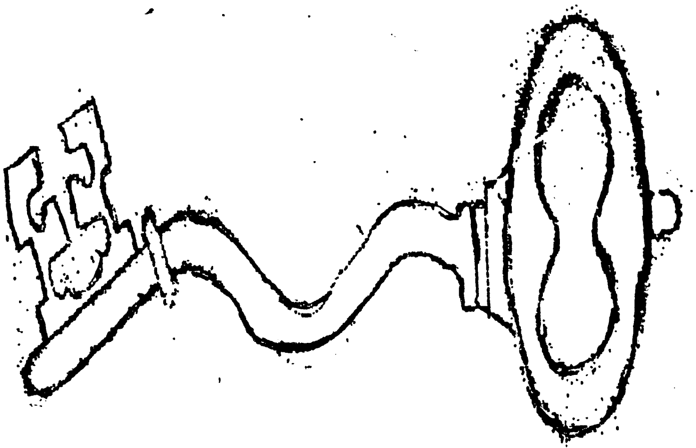
When you have finished, type CLEAR to stabilise the pointers, then list your program. You will find the words you wish to be tokens on a single DATA line (4990) then each tokenised description ten lines apart starting from line 5000. You can change this if you want to. Some of the lines will be longer than 64 characters and for this reason you won't be able to edit some lines.

Next delete all of the program except for the data statements by typing:

```
0 # 10 - 3300
POKE 31469,182
RUN
```

Then delete line 0.

You will then have a set of room descriptions which may be saved to tape or disk. You can now add the rest of your program.



This is the "key" to the developement of the microprocesor.

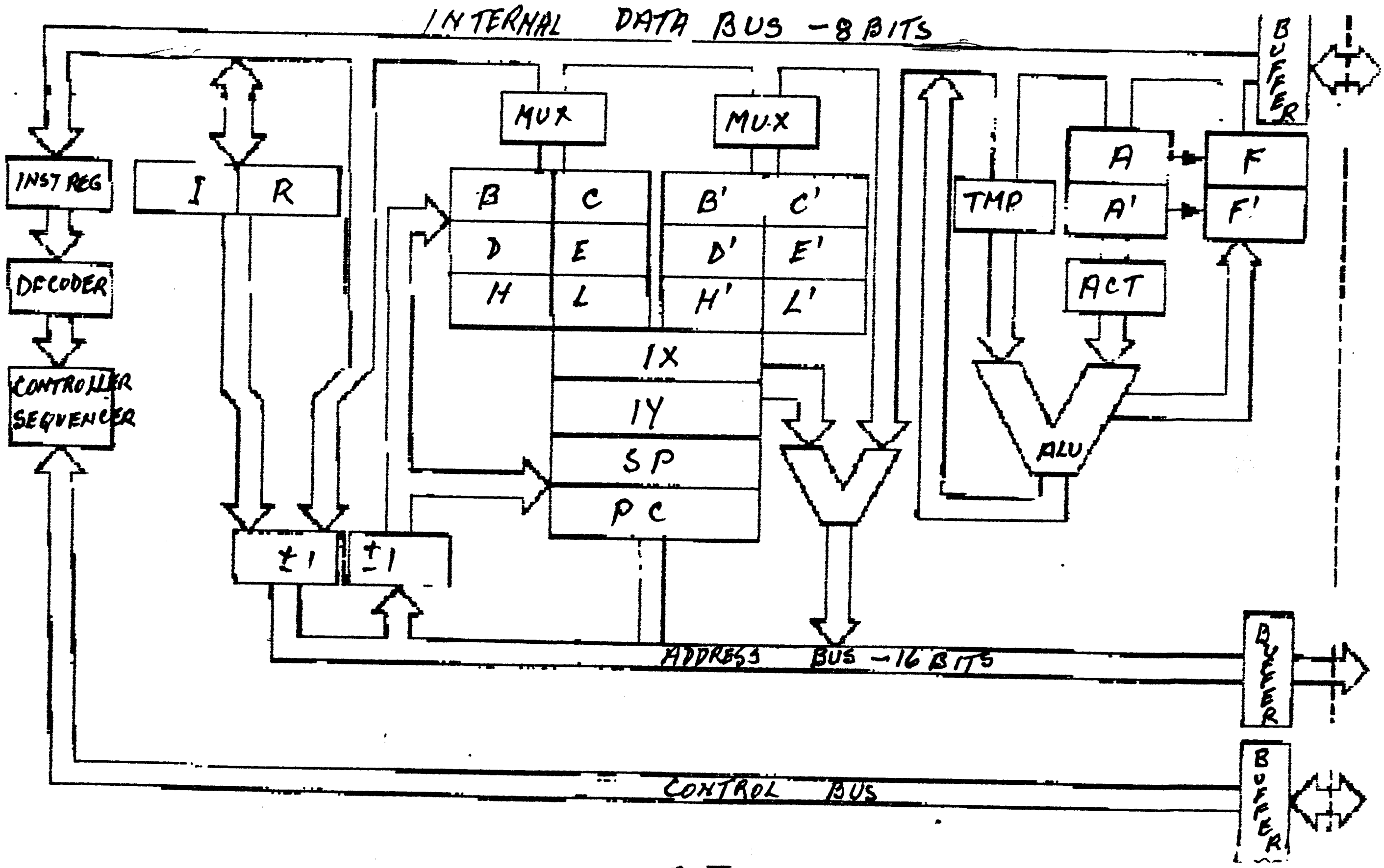
In early 1971 the only two standard LSI products were the first 1K-bit dynamic ram and the UART (receiver/transmitter chip) had been developed. It was then that the microprocessor was accidientially introduced. The introduction in 1971 of the Intel 4004, a 4-bit PMOS microprocessor resulted from a contract with a Japanese calculator manufacturer. They had to agree not to develope a calculator for one year. This chip was not powerful enough for a computer, being designed as a "general purpose microprocessor" for calculators. High sales were not expected, but they occurred. The next significant event was in 1972 when Intel introduced the 8008, the first general purpose 8 bit microprocessor. Display Terminals Corporation, now known as Datapoint, then a CRT display manufacturer, requested bids for the production of a monalithic processor capable of controlling a CTR. Two companies, Texas Instruments and Intel vied for and obtained the development contracts. After months of effort, Texas Instruments withdrew. Intel continued the development and came up with a component that could satisfy all but one of Datapoint's functional requirements; it was too slow. At about the same time, a price war had started over the prices of bi-polar devices. For these reasons Datapoint decided to implement it's CRT controller in bi-polar technecology. Intel, then a very young company, was left with a chip, whose development had been paid for, but for which there was no obvious market. Since Intel produced memory products, the 8008 was introduced on the market on the assumption that it would sell memory chips. All design efforts were halted and the design team were assigned to other tasks. That was to be the end of microprocessors at Intel.

To the surprise of it's manufacturers (and of it's competitors) sales of the microprocessor progressed rapidly. Intel had stumbled, by accident, on the next standard LSI product, the microprocessor. Intel quickly realised the potential of the device, reassembled a design team and a year later introduced the successor to the 8008, the 8080. Simultaneously Intel's competitors set to work on their version of what the 8080 should have been had it been designed correctly for it's market. Within two years all the main "standard" 8-bit microcomputers had been introduced, most of them inspired by the early design of the 8080. Motorola introduced the 6800 nearly one year after the 8080, Rockwell

the pps-8.Signetics the 2650 etc.

In the third generation the succors to the 8080 were introduced. The Z80 from Zilog, the 8085 from Intel, and the 6809 from Motorola, and the first 1-chip microcomputers the F8 from Fairchild and Mostek and the 8048 from Intel, the TMS1000 and the9940 from Texas Instruments.

Three of the 8080 designers left Intel and created their own company, ZILOG. The story of Zilog is similar to the story of most Silicon Valley companies. Started by a small group of engineers Zilog has grown to an important semiconductor company. It's first product was the Z80 designed to compete with the 8080. The Z80incorporates the 8080,the 8224 clock and the 8228 system controller, and some additional facilities on a single chip. The Z80 instruction set is compatiable with the 8080 set. An 8080 ROM can run "as is" on the Z80 system, and the Z80 has some additional instructions. The standard Z80 is as fast as the fast 8080. A faster version the Z80A operates at 4MHz. The internal organization of the Z80 is shown below. It closely follows the 8080. It provies however one main inprovement. It is equipped with two banks of registers. These registers can be used to provide either a large number of internal registers or a very fast response to a single level of interrupt. These banks are implanted correctly for interrupt handling. The accumulator and status registers are also duplicated. The Z80 is also equipped with a memory refresh facility that allows certain dynamic RAMS to be connected directly to the system without the need for an external refresh circuit.



```
10 '            *************************************
20 '            ***   DISPLAY INVERSE CHARACTER   ***
30 '            ***          SET IN ROM           ***
40 '            ***     AS USED BY DOT MATRIX     ***
50 '            ***            PRINTER            ***
60 '            ***    R. B. KITCH   27/9/86      ***
70 '            *************************************
80 '
100 'WHEN INVERSE CHARACTERS ARE SENT TO A DOT MATRIX PRINTER
110 'THE PRINTER SHIFTS TO GRAPHICS MODE AND REQUIRES A ROUTINE
120 'TO SUPPLY THE APPROPRIATE SHAPES TO THE HEAD. (NORMAL
130 'CHARACTERS ARE HELD IN THE PRINTERS ROM)
140 'IN THE VZ COMPUTER A TABLE OF SHAPES IS LOCATED AT
150 '3B94H TO 3CD3 IN ROM. THERE ARE 64 CHARACTERS, EACH USING
160 '5 BYTES TO DEFINE THEIR GRAPHIC SHAPE. THE SHAPES MAY BE
170 'DECODED AND OUTPUT TO THE SCREEN AS IS DONE IN THIS
180 'PROGRAM. NOTE THAT THERE ARE SOME ERRORS IN THE ROM.
190 'THE 5 BYTES DEFINE A 5 BY 8 DOT MATRIX WHICH IS THE SHAPE
200 'OF THE CHARACTER, WHICH INCIDENTLY ARE NOT ORDERED
210 'ACCORDING TO THE ASCII CODE.
220 'THE FIRST BYTE DEFINES THE LEFT HAND EDGE OF THE CHARACTER-
230 'WHICH IS THE FIRST PRINTED DURING A PASS OF THE PRINTER
240 'HEAD. IN TANDY PRINTERS THE MSB IS THE LOWERMOST PIN OF THE
250 'HEAD AND THE LSB IS THE UPPERMOST PIN. THE PINS ON EPSON
260 'PRINTER HEADS ARE ARRANGED IN THE OPPOSITE SENSE. THIS
270 'REQUIRES THAT THE BITS IN EACH BYTE BE REVERSED.
280 '*************************************************************
290 '
300 DIM MK%(7)  :'***VECTOR OF BIT MASK VALUES - POWERS OF 2
310 DIM BT%(7)  :'***VECTOR OF DECODED BITS FROM ROM VALUE.
320 '
330 '***FILL MASK VECTOR WITH POWERS OF 2 FOR DECODING.
340 FOR I%=0 TO 7 :MK%(I%)=2^I% :NEXT I%
350 '
400 '***INITIALIZE PARAMETERS - MAY BE CHANGED TO VARY SCREEN.
410 CC%=4      :'***CHARACTER COLOUR.  (1-4)
420 BC%=2      :'***BACKGROUND COLOUR. (1-4)
430 CS%=0      :'***COLOUR SET. (0-1)
440 CW%=3      :'***COLUMN WIDTH BETWEEN CHARACTERS.
450 SP%=16     :'***ROW SPACING FOR CHARACTERS.
460 HS%=0      :'***STARTING HORIZONTAL POSITION ON HI-RES SCREEN.
470 VP%=3      :'***STARTING VERTICAL POSITION ON HI-RES SCREEN.
480 HM%=127    :'***MAXIMUM HORIZONTAL POSITION. (0-127)
490 '
600 '***SET UP MAIN LOOP TO STEP THROUGH ROM FROM 3B94H-3CD3.
610 BK%=0                     :'***BYTE COUNTER FOR EACH CHARACTER.
620 HP%=HS%                   :'***SET HORIZONTAL POSITION TO START
630 MODE(1) :COLOR,CS%        :'***SET HI-RES SCREEN AND COLOR SET.
640 FOR AD%=15252 TO 15571    :'***ROM ADDRESSES FOR SHAPE TABLE.
650   DV%=PEEK(AD%)           :'***DECIMAL VALUE IN ROM.
```

```
660 '
700 '***DECODE THE INDIVIDUAL BITS OF DV% AND STORE IN BT%().
710 '***THE MASK VALUES IN MK%() ARE "ANDED" WITH THE VALUE.
720 '***THE RESULT STORED IN BT%() IS THE "COLOUR" OF THE BIT.
730   FOR I%= 0 TO 7          :'***PROCEED FROM LSB TO MSB.
740     IF DV% AND MK%(I%) THEN BT%(I%)=BC% ELSE BT%(I%)=CC%
750   NEXT I%
800 '
810 '***CHECK THAT THERE IS ENOUGH ROOM TO PLOT CHARACTER.
820   IF BK%=0 AND HM%-HP%<4 THEN HP%=HS% :VP%=VP%+SP%:'*NEW ROW
830   BK%=BK%+1               :'***INCREMENT BYTE COUNTER.
840 '
900 '***OUTPUT BYTE TO SCREEN.
910   FOR I%=0 TO 7
920     COLOR BT%(I%)         :'***SET COLOUR OF BIT.
930     SET(HP%,VP%+I%)       :'***PLOT BIT.
940   NEXT I%
950 '
1000 '***PREPARE FOR NEXT BYTE.
1010   HP%=HP%+1              :'***INCREMENT HORIZONTAL POSITION.
1020   IF BK%=5 THEN BK%=0 :HP%=HP%+CW%       :'***NEW CHARACTER.
1030 NEXT AD%
2000 GOTO 2000 :END
```

TRADING POST

FOR SALE.

PRINTER. GP100. Dick Smith Graphic Printer. This model is completely conpatiable with the VZ. They were specially set-up for the VZ, and will produce INVERSE and GRAPHIC characters and copy the mode 1 and o screens. PRICE $120.

PRINTER. SEIKOSHA GP250X graphic printer. This printer will print all ASCII characters from the VZ, and though it is a graphic printer we do not have a patch to make it compatiable with the VZ graphics or inverse characters. It will copy the Mode 0 screen, but not the Mode 1. It does however have a full forte of it's own graphics. It will also print double height-double width characters. I don't doubt that a patch could be written for it. PRICE $100 or reasonable offer.

Postage will depend on where you live. The weight is about 5KG. Both have manuals for them. Insurance is $3.

Get in touch with me. Harry.

OTHER V Z USER GROUPS

H.V.V.Z.U.G
P.O.Box 161
JESMOND NSW.2299.

DISKMAG
P.O.Box 600.
Taree NSW. 2430.

CENT.VIC.COMP.Club
24 Breen St.
BENDIGO VIC 3550

BRISBANE VZUG
63 Tingalpa St.
WYNUM West. Q'ld. 4178

Graeme Bywater
P.O.Box 388
MORLEY W.A. 6062